SU3lib: A C++ library for accurate computation of Wigner and Racah coefficients of SU(3)

Tomáš Dytrych[a,b,*], Daniel Langr[c], Jerry P. Draayer[b], Kristina D. Launey[b], Daniel Gazda[a]

[a] *Department of Physics and Astronomy, Louisiana State University, Baton Rouge, LA 70803, USA*
[b] *Nuclear Physics Institute, Czech Academy of Sciences, 25068 Řež, Czech Republic*
[c] *Department of Computer Systems, Faculty of Information Technology, Czech Technical University in Prague, 16000 Praha, Czech Republic*

Abstract

We present the C++ library SU3lib for accurate computation of SU(3) Wigner coupling and Racah recoupling coefficients. It is built on the efficient mathematical algorithm originally proposed by Draayer and Akyiama [1, 2]. Our implementation allows for highly accurate computations of SU(3) ⊃ SO(3) Wigner coefficients, including large SU(3) irreducible representations (irreps) that were heretofore inaccessible due to the loss of precision encountered in series of large alternating terms. As large SU(3) irreps play an important role in medium- and heavy-mass atomic nuclei, SU3lib expands the scope of approaches to nuclear structure and reactions that rely on available SU(3) coupling-recoupling coefficients.

Keywords: SU(3), SU(2), SO(3), Wigner coefficients, Racah coefficients, Coupling coefficients, Isoscalar factors

PROGRAM SUMMARY

Program Title: su3lib
CPC Library link to program files: (to be added by Technical Editor)
Developer's repository: https://gitlab.com/tdytrych/su3lib
Code Ocean capsule: (to be added by Technical Editor)
Licensing provisions(please choose one): MIT
Programming language: C++
External libraries: WIGXJPF [3], Boost
Nature of problem: Accurate calculation of SU(3) ⊃ SO(3) and SU(3) ⊃ SU(2) × U(1) Wigner coupling and Racah recoupling coefficients for arbitrary couplings and multiplicity.
Solution method: We adopt the mathematical procedure proposed by Draayer and Akyiama [1], who also provided its implementation as a FORTRAN library [2]. The challenge is to avoid the loss of precision due to cancellation in sums of large alternating terms in transformation between SU(3) ⊃ SO(3) and SU(3) ⊃ SU(2) × U(1) schemes. The present library achieves this through implementing key formulas and data structures as C++ templates and utilizing floating-point data types with extended precision provided by the `Boost.Multiprecision` library as template arguments. This permits an efficient and accurate computation of SU(3) coefficients even for large SU(3) irreps that were heretofore inaccesible.
Additional comments including restrictions and unusual features (approx. 50-250 words):

[1] J. P. Draayer and Y. Akiyama, *J. Math. Phys.* **14**, 1904 (1973).
[2] Y. Akiyama and J. P. Draayer, *Comput. Phys. Commun.* **5**, 405 (1973).
[3] H. T. Johansson and C. Forssn, SIAM J. Sci. Comput. **38**(1), A376 (2016).

1. Introduction

The symmetry group SU(3) and its representation theory play an important role in elementary particle physics [1, 2] and nuclear physics [3]. In nuclear physics, the importance of SU(3) group stems from the fact that it emerges

*Corresponding author.
E-mail address: tdytrych@phys.lsu.edu

as a physically relevant subgroup of many successful group-theoretical models of atomic nuclei [4, 5]. For example, the relevance of SU(3) for the shell model description of rotating deformed nuclei has been established early on by the seminal work of Elliott [3]. The importance of SU(3) was further reinforced by the fact that it is also a subgroup the symplectic model [6, 7] that provides a comprehensive theoretical framework for understanding deformation-dominated nuclear collective dynamics and emergent phenomenon of the nuclear shape coexistence [8].

The practical usage of SU(3) symmetry group and its representations in quantum physics rely on accurate and efficient computations of SU(3) Wigner coupling and Racah recoupling coefficients. Wigner coefficients are given either in canonical SU(3) $\supset$ SU(2) $\times$ U(1) group chain basis, which is of interest in particle physics, or in physical SU(3) $\supset$ SO(3) group chain basis that is relevant for the description of many-nucleon system as the total orbital angular momentum L is a good quantum number in this scheme. The physical basis states also bear an additional multiplicity label that reflects the fact that multiple occurrences of L are possible within a generic irreducible representation (irrep) of SU(3). The construction of physical basis by the Elliotts projection operator method [3] is more involved. Recently, however, some progress was made in simplifying this approach [9]. In addition, SU(3) Wigner coefficients also depend on the outer multiplicity label, which is needed to distinguish multiple equivalent SU(3) irreps that occur in the tensor product space of two SU(3) irreps. Although there exist a plethora of work dedicated to determining SU(3) coupling-recoupling coefficients (see references in [10] and recently proposed method [11]), only two publicly available libraries provide Wigner coupling coefficients in both canonical and physical basis [12, 13].

The first comprehensive method for the computation of SU(3) coupling-recoupling coefficients for arbitrary couplings and multiplicities was originally proposed by Draayer and Akyiama [14] and implemented in form of efficient Fortran 77 subroutines [12]. This method resolves the outer multiplicity via the recursive prescription that yields particularly simple symmetry properties of Wigner coefficients [14]. An alternative method to determine SU(3) Wigner coefficients based on coherent state theory was proposed by Rowe and Bahri [10] and implemented as a set of Fortran 77 subroutines [13], which introduced some new techniques in the evaluation of alternating series that improve efficiency and accuracy of the algorithm for the computation of SU(3) $\supset$ SO(3) Wigner coefficients. This approach resolves the outer multiplicity problem non-recursively by solving a system of linear equations leading to outcomes that are not identical to those of ref. [14] beyond the simplest cases where the outer multiplicity is equal to one.

The aim of the present article is to provide a description of a new comprehensive C++ library that extend the reach of Draayer and Akyiama method [14] beyond its current limits towards larger SU(3) irreps and larger outer and inner multiplicities that were heretofore numerically inaccessible due to the loss of precision encountered in computing series of large alternating terms. As large SU(3) irreps and higher multiplicities arise naturally in nuclear systems beyond ^{16}O, the present library extends the reach of the *ab initio* symmetry-adapted no-core shell model framework [15, 16] for large-scale studies of nuclear structure and reactions [17, 18, 19, 20, 21]. At the same time, it is also of practical usage for other models of atomic nuclei that rely on availability of accurate SU(3) coefficients.

2. Library interface

The SU3lib provides a very simple application programming interface (API) exposed by the header file `su3.h`. Its main functions are shown in Table 1. They are organised in the namespace `su3` and adhere to the naming convention used in the legacy `FORTRAN` implementation [12].

2.1. Initialization and finalizing routines

The initial setup of SU3lib for single-threaded applications is provided by the function `su3::init`($M_{\max}, 2j_{\max}$). The input parameter $M_{\max}$ defines the allowed range of SU(3)$\supset$SO(3) quantum numbers $(\lambda\,\mu)$ and L, which must satisfy $\lambda + \mu + L \leq M_{\max}$. Internally, $M_{\max}$ sets the range of binomial coefficients $\binom{k}{l}$, $l \leq k \leq M_{\max}$, that are generated during the initial setup and which are used for the computation of SU(3)$\supset$SO(3) Wigner coefficients. If not provided, $M_{\max}$ value is set implicitly to 200. The second input parameter $2j_{max}$ is used for the initialization of WIGXJPF library [22]. It represents the maximal argument for the computation of $3j$ and $6j$ symbols. If not provided, its value is set to $2j_{\max} = M_{\max}$. For all practical purposes, both implicitly set values of $M_{\max}$ and $2j_{\max}$ are sufficiently high so as to impose no serious limitations. If `DTU3R3_CACHE` is defined, then an internal cache is created for storing transformation coefficients between the Cartesian and spherical SU(3) basis states. Multi-threaded initialization is implemented by the function `su3::init_thread`($M_{\max}, 2j_{\max}$). It must be executed by each thread that will call

Table 1: Main functions of the `su3` namespace.

Task/Computation	Function name in `su3` namespace
initialization of SU3lib	`init`$(M_{\max}, 2j_{\max})$
	`init_threaded`$(M_{\max}, 2j_{\max})$
cleanup of SU3lib	`finalize()`
	`finalize_threaded()`
generate SU(3) tensor product space $(\lambda_1\,\mu_1) \otimes (\lambda_2\,\mu_2) \rightarrow \sum_{\oplus} \rho^{\max}(\lambda\,\mu)$	`couple`$(\lambda_1, \mu_1, \lambda_2, \mu_2, \vec{l})$
multiplicity for $(\lambda_1\,\mu_1) \otimes (\lambda_2\,\mu_2) \rightarrow (\lambda_3\,\mu_3)$ coupling	`mult`$(\lambda_1, \mu_1, \lambda_2, \mu_2, \lambda_3, \mu_3)$
multiplicity of L in $(\lambda\,\mu)$	`kmax`(λ, μ, L)
SU(3) $\supset$ SO(3) Wigner coefficients	`wru3r3`
SU(3) $\supset$ SU(2) $\times$ U(1) Wigner coefficients	`xwu3`
$U6$-$(\lambda\,\mu)$ coefficients	`wru3`
$Z6$-$(\lambda\,\mu)$ coefficients	`wzu3`
9-$(\lambda\,\mu)$ coefficients	`wu39lm`

SU3lib and WIGXJPF API. When the API functions are no longer needed, the internal memory can be cleared by `su3::finalize` and `su3::finalize_threaded` for single and multi-threaded applications, respectively.

2.2. Utility functions

Besides computing SU(3) coupling-recoupling coefficients, the library also provides some basic utility functions.

- `su3::couple`$(\lambda_1, \mu_1, \lambda_2, \mu_2, \vec{l})$ invokes SU(3) coupling rules to determine a list of all the possible SU(3) irreps $(\lambda\,\mu)$ and their maximal multiplicities $\rho^{\max}$ that span the tensor product space of two SU(3) irreps $(\lambda_1\,\mu_1) \otimes (\lambda_2\,\mu_2)$. The symbol $\vec{l}$ signifies the resulting list represented as `std::vector<std::tuple<`$\rho^{\max}$`,`λ`,`μ`>>` data type.

- `su3::mult`$(\lambda_1, \mu_1, \lambda_2, \mu_2, \lambda, \mu)$ returns multiplicity $\rho^{\max}$ with which an irrep $(\lambda\,\mu)$ of SU(3) occurs in the tensor product space $(\lambda_1\,\mu_2) \otimes (\lambda_2\,\mu_2)$.

- `su3::kmax`(λ, μ, L) returns multiplicity $k^{\max}$ of a given orbital angular momentum value L in $(\lambda\,\mu)$ irrep. Returns zero if $L \notin (\lambda\,\mu)$.

2.3. Calculation of SU(3) *coupling-recoupling coefficients*

A common feature of all the API functions for computing SU(3) coupling-recoupling coefficients is the arrangement of resulting coefficients as a linear array, which we denote $\vec{d}$, represented by `std::vector<double>` data type. The particular order of coefficients in memory for a given type of coupling-recoupling coefficient is decribed in detail for each API function.

- `su3::wru3r3`$(\lambda_1,\ \mu_1,\ \lambda_2,\ \mu_2,\ \lambda_3,\ \mu_3,\ L_1,\ L_2,\ L_3,\ \rho^{\max},\ k_1^{\max},\ k_2^{\max},\ k_3^{\max},\ \vec{d})$ computes SU(3) $\supset$ SO(3) Wigner coefficients $\langle (\lambda_1\,\mu_1) k_1 L_1; (\lambda_2\,\mu_2) k_2 L_2 \| (\lambda_3\,\mu_3) k_3 L_3 \rangle_\rho$ for $0 \le \rho < \rho_{\max}$, $0 \le k_1 < k_1^{\max}$, $0 \le k_2 < k_2^{\max}$, and $0 \le k_3 < k_3^{\max}$, where $\rho^{\max}$ denotes the multiplicity of $(\lambda\,\mu)$ irrep in the tensor product space $(\lambda_1\,\mu_2) \otimes (\lambda_2\,\mu_2)$ and $k_i^{\max}$ represents the number of occurrences of a given L_i in the $(\lambda_i\,\mu_i)$ irrep. The resulting Wigner coefficients are stored in vector $\vec{d}$ in the following order:

$$d_i = \langle (\lambda_1\,\mu_1) k_1 L_1; (\lambda_2\,\mu_2) k_2 L_2 \| (\lambda_3\,\mu_3) k_3 L_3 \rangle_\rho, \text{ where } i = \rho + \rho^{\max}\left(k_1 + k_1^{max}\left(k_2 + k_2^{\max} k_3\right)\right).$$

- `su3::xwu3`$(\lambda_1,\ \mu_1,\ \lambda_2,\ \mu_2,\ \lambda_3,\ \mu_3,\ \epsilon_3,\ 2\Lambda_3,\ \rho^{\max},\ \vec{l},\ \vec{d})$ computes SU(3) $\supset$ SU(2) $\times$ U(1) Wigner coefficients $\langle (\lambda_1\,\mu_1) \epsilon_1 \Lambda_1; (\lambda_2\,\mu_2) \epsilon_2 \Lambda_2 \| (\lambda_3\,\mu_3) \epsilon_3 \Lambda_3 \rangle_\rho$. Coefficients are computed for all values of the outer multiplicity ρ ($0 \le \rho < \rho^{\max}$), and for all combinations of SU(2) $\times$ U(1) labels $\epsilon_1\Lambda_1 \in (\lambda_1\,\mu_1)$ and $\epsilon_2\Lambda_2 \in$

$(\lambda_2\,\mu_2)$ that couple to the given labels $\epsilon_3\Lambda_3$. The symbol $\vec{l}$ denotes the resulting list of labels implemented as `std::vector<std::tuple<`ϵ_2`,`$2\Lambda_2$`,`$2\Lambda_1$`>>` data type. Note that ϵ_1 is not provided explicitly as $\epsilon_1 = \epsilon_3 - \epsilon_2$. The order of resulting coefficients $\vec{d}$ follows the order of labels given in $\vec{l}$, that is

$$\begin{aligned} l_i &\rightarrow \epsilon_1\,\Lambda_1\,\epsilon_2\Lambda_2, \\ d_j &= \langle((\lambda_1\,\mu_1)\epsilon_1\Lambda_1;(\lambda_2\,\mu_2)\epsilon_2\Lambda_2\|(\lambda_3\,\mu_3)\epsilon_3\Lambda_3\rangle_\rho, \end{aligned}$$

where the index $j = i\rho^{\max} + \rho$.

- `su3::wru3`(λ_1, μ_1, λ_2, μ_2, λ, μ, λ_3, μ_3, λ_{12}, μ_{12},λ_{23}, μ_{23}, $\rho_{12}^{\max}$, $\rho_{12,3}^{max}$, $\rho_{23}^{\max}$, $\rho_{1,23}^{\max}$, $\vec{d}$) computes $U6$-$(\lambda\,\mu)$ coefficients for all four outer multiplicities,

$$\begin{aligned} &0 \le \rho_{12} < \rho_{12}^{\max} : (\lambda_1,\mu_1)\otimes(\lambda_2,\mu_2) \rightarrow (\lambda_{12}\,\mu_{12}) \\ &0 \le \rho_{12,3} < \rho_{12,3}^{\max} : (\lambda_{12}\,\mu_{12})\otimes(\lambda_3\,\mu_3) \rightarrow (\lambda,\mu) \\ &0 \le \rho_{23} < \rho_{23}^{\max} : (\lambda_2\,\mu_2)\otimes(\lambda_3\,\mu_3) \rightarrow (\lambda_{23}\,\mu_{23}) \\ &0 \le \rho_{1,23} < \rho_{1,23}^{\max} : (\lambda_1\,\mu_1)\otimes(\lambda_{23}\,\mu_{23}) \rightarrow (\lambda,\mu). \end{aligned}$$

  Resulting coefficients are given in the following order:

$$d_i = U\left[(\lambda_1\,\mu_1)(\lambda_2\,\mu_2)(\lambda\,\mu)(\lambda_3\,\mu_3);(\lambda_{12}\,\mu_{12})\rho_{12}\rho_{12,3}(\lambda_{23}\,\mu_{23})\rho_{23}\rho_{1,23}\right],$$

  where the index i for given multiplicities $\rho_{1,23},\rho_{23},\rho_{12,3},\rho_{12}$ is equal to

$$i = \rho_{12} + \rho_{12}^{\max}\left(\rho_{12,3} + \rho_{12,3}^{\max}\left(\rho_{23} + \rho_{23}^{\max}\rho_{1,23}\right)\right).$$

- `su3::wzu3`(λ_1, μ_1, λ_2, μ_2, λ, μ, λ_3, μ_3, λ_{12}, μ_{12},λ_{23}, μ_{23}, $\rho_{12}^{\max}$, $\rho_{12,3}^{max}$, $\rho_{23}^{\max}$, $\rho_{1,23}^{\max}$, $\vec{d}$) computes $Z6$-$(\lambda\,\mu)$ coefficients [23]. The outer multiplicities and the layout of resulting coefficients in memory is identical to `su3::wru3`.

- `su3::wu39lm`(λ_1, μ_1, λ_2, μ_2, λ_{12}, μ_{12}, λ_3, μ_3, λ_4, μ_4, λ_{34}, μ_{34}, λ_{13}, μ_{13}, λ_{24}, μ_{24}, λ, μ, $\vec{d}$) computes 9-$(\lambda\,\mu)$ coefficients for all six multiplicities

$$\begin{aligned} &0 \le \rho_{12} < \rho_{12}^{\max} : (\lambda_1,\mu_1)\otimes(\lambda_2,\mu_2) \rightarrow (\lambda_{12}\,\mu_{12}) \\ &0 \le \rho_{34} < \rho_{34}^{\max} : (\lambda_3\,\mu_3)\otimes(\lambda_4\,\mu_4) \rightarrow (\lambda_{34},\mu_{34}) \\ &0 \le \rho_{13,24} < \rho_{13,24}^{\max} : (\lambda_{13}\,\mu_{13})\otimes(\lambda_{24}\,\mu_{24}) \rightarrow (\lambda_{13,24}\,\mu_{13,24}) \\ &0 \le \rho_{13} < \rho_{13}^{\max} : (\lambda_1\,\mu_1)\otimes(\lambda_3\,\mu_3) \rightarrow (\lambda_{13}\,\mu_{13}) \\ &0 \le \rho_{24} < \rho_{24}^{\max} : (\lambda_2\,\mu_2)\otimes(\lambda_4\,\mu_4) \rightarrow (\lambda_{24}\,\mu_{24}) \end{aligned}$$

  Note that in this case a user does not need to provide multiplicities as the function `su3::wu39lm` determines them internally. The order of 9-$(\lambda\,\mu)$ coefficients in memory is the following:

$$d_i = \begin{Bmatrix} (\lambda_1\,\mu_1) & (\lambda_2\,\mu_2) & (\lambda_{12}\,\mu_{12}) & \rho_{12} \\ (\lambda_3\,\mu_3) & (\lambda_4\,\mu_4) & (\lambda_{34}\,\mu_{34}) & \rho_{34} \\ (\lambda_{13}\,\mu_{13}) & (\lambda_{24}\,\mu_{24}) & (\lambda\,\mu) & \rho_{13,24} \\ \rho_{13} & \rho_{24} & \rho_{12,34} & \end{Bmatrix},$$

  where the index i for given multiplicities $\rho_{13,24},\rho_{24},\rho_{13},\rho_{12,34},\rho_{34},\rho_{12}$ is equal to

$$i = \rho_{12} + \rho_{12}^{\max}\left(\rho_{34} + \rho_{34}^{\max}\left(\rho_{12,34} + \rho_{12,34}^{\max}\left(\rho_{13} + \rho_{13}^{\max}\left(\rho_{24} + \rho_{24}^{\max}\rho_{13,24}\right)\right)\right)\right).$$

3. Implementation details

A detailed description of the algebraic foundations upon which the codes are based is available in ref. [14]. One of the main challenges is the accurate evaluation of SU(3) $\supset$ SO(3) Wigner coefficients. They are obtained from the corresponding SU(3) $\supset$ SU(2) $\times$ U(1) Wigner coefficients via a unitary transformation given by the overlaps between the Cartesian SU(3) $\supset$ SU(2) $\times$ U(1) and spherical SU(3) $\supset$ SO(3) basis states. The analytical expression for this unitary transformation is provided by the relation (26) in [14]. It involves series of large alternating terms causing diminishing accuracy with increasing SU(3) quantum numbers $(\lambda\,\mu)$ and increasing L values. For example, transformation coefficients computed with double precision for SU(3) irrep (20 0), which is important, e.g., for the description of the Hoyle state in ^{12}C [24], yield relative errors ranging from 10^{-14} for $L = 0$ up to 10^{-4} for $L = 12$. Our solution to this challenge is twofold.

First, we adopt a very simple yet effective strategy presented in [13]. The idea is to improve both accuracy and efficiency by reducing the number of terms in the alternating sum

$$I(p,q,\sigma) = \sum_{n=\max(0,\sigma-p)}^{\min(q,\sigma)} (-1)^n \binom{q}{n}\binom{p}{\sigma-n} \tag{1}$$

that is central to the computation of overlaps between Cartesian and spherical basis states. The integer function $I(p,q,\sigma)$ can be expressed using an alternative form

$$I(p,q,\sigma) = (-1)^\sigma \sum_{n=\max(0,\lceil(\sigma+p-q)/2\rceil)}^{\min(p,\lfloor\sigma/2\rfloor)} (-1)^n \binom{p}{n}\binom{q-p}{\sigma-2n}, \tag{2}$$

for $p \geq q$ with the simple symmetry relation $I(p,q,\sigma) = (-1)^\sigma I(q,p,\sigma)$ for $p < q$. As can be seen from ranges of the sums, the latter expression effectively cuts the number of terms in half. As illustrated in [13], this improves accuracy for large values of input arguments p, q and σ that arise in the case of large SU(3) irreps.

Second, the transformation is implemented as a C++ function template operating on data structures that store binomial coefficients and their inverses with floating point precision required to avoid the loss of accuracy. This generic feature of C++ enables us to carry out computations of transformation coefficients in double/extended double/quadruple/octuple precision, and, if needed, to readily extend accuracy by employing higher precision data types. The floating point data types which are not native on a given CPU architecture are provided by the Boost.Multiprecision library. The appropriate floating-point precision for a given overlap is set based on $\lambda + \mu + L$ value and the relative accuracy better than 5×10^{-13} up to $\lambda + \mu + L \leq 268$ is guaranteed.

The computation of a large set of SU(3) $\supset$ SO(3) Wigner coefficients can be further accelerated if the overlaps between SU(3) Cartesian and spherical basis, which represent the most computationally expensive part of the algorithm, are stored in memory in a fast hash table represented by `std::unordered_map` C++ type. Enabling this feature is described in the next section.

Computation of SU(3) coupling-recoupling coefficients involves a frequent use of SU(2) $3j$ and $6j$ symbols. Their computation may suffer from an increasing loss of accuracy commonly experienced for large input angular momenta [22] that arise in large SU(3) irreps. To reduce such an error we adopted WIGXJPF library [22]. This library implements a fast algorithm for evaluation of any SU(2) recoupling coefficient with a fixed relative accuracy 6.66×10^{-16} that extends to very large values of ingoing angular momenta.

4. Installation and setup

4.1. Dependencies

SU3lib requires the following tools and libraries:

1. A C++ compiler conforming to the C++11 Standard or newer.
2. WIGXJPF library provided by H.T. Johansson from the Chalmers University of Technology, Sweden [22].
3. Boost library, namely its Boost.Multiprecision and Boost.Math sub-libraries.

4.2. Configuration

The configuration of the SU3lib building process is defined by the `config.mk` file, which respects the syntax of makefiles. This configuration consists of the following items:

- `BOOST_ROOT` — the root directory of the Boost library. May be provided in the form of the environment variable, or defined directly in `config.mk`.
- `WIGSIXJPFDIR` — the root directory of the WIGXJPF library. May be provided in the form of the environment variable, or defined directly `config.mk`.
- `CXX` — specifies the name of the C++ compiler and linker.
- `CXXFLAGS` — compiler flags, such as optimization and debugging options.
- `CXXFLAGS` — linker flags, such as specifications of linked libraries.
- `DTU3R3_CACHE` — C++ preprocessor symbol enabling the support for caching of transformation coefficients between the Cartesian and spherical basis schemes of SU(3).
- OpenMP support — when enabled, SU3lib will be built with the support of OpenMP threading.
- 80-bit `long double` support – when enabled, the `long double` data type is used for extended precision (80-bit) floating-point calculations. This is suitable if a C++ implementation is able to utilize the hardware support for extended precision floating-point operations. (This is the case, for example, of most compilers on Linux x86/x86_64 systems, where x87 floating-point instructions are issued.) When disabled, the `cpp_bin_float_double_extended` data type from Boost.Multiprecision is used instead.
- `float128` support — when enabled, SU3lib will be built with the support for `float128` data type provided by GNU libquadmath, which may accelerate quadruple-precision computations. When disabled, the `cpp_bin_float_quad` data type from Boost.Multiprecision is used instead.

4.3. Installation

After the configuration for a particular system has been set, the library and its tools may be built with the `make all` command. The `make install` command will then install the created binary library file into the `lib` subdirectory. By default, the resulting library file will be `lib/libsu3.a`. The corresponding header files for this library are available in the `include` directory.

SU3lib also provides simple programs for computations of SU(3) coupling-recoupling coefficients. They also serve as examples how to use library and utility functions shown in Table 1. The associted source codes can be found in `tools` subdirectory. Executables are built by `make tools` command.

5. Summary

We introduced a C++ library for the computation of SU(3) coupling-recoupling coefficients. The present library extends significantly the reach of the underlying method [14] towards large SU(3) irreps and outer multiplicities, that were heretofore numerically inaccessible. As large SU(3) irreps are needed for the description of nuclear structure beyond the light nuclei, SU3lib represents a major enabling feature for approaches to nuclear structure and reactions that rely on available SU(3) coupling-recoupling coefficients.

Acknowledgements

The work was supported in part by the Czech Ministry of Education, Youth and Sports under Grant No. CZ.02.-1.01/0.0/0.0/16_019/0000765, the U.S. National Science Foundation (OIA-1738287, ACI-1713690, PHY-1913728) and SURA.

[1] Y. Ne'eman, Derivation of strong interactions from a gauge invariance, Nucl. Phys. 26 (1961) 222.
[2] M. Gell-Mann, Symmetries of Baryons and Mesons, Phys. Rev. 125 (1962) 1067.
[3] J. P. Elliott, Collective motion in the nuclear shell model. I. Classification schemes for states of mixed configurations, Proc. R. Soc. A 245 (1240) (1958) 128–145. doi:10.1098/rspa.1958.0072.
[4] D. J. Rowe, J. L. Wood, Fundamentals Of Nuclear Models: Foundational Models, World Scientific, Singapore, 2010.
[5] F. Iachello, A. Arima, The Interactiong Boson Model, Cambridge University Press, 1987. doi:10.1017/CBO9780511895517.
[6] G. Rosensteel, D. J. Rowe, Nuclear Sp(3,R) model, Phys. Rev. Lett. 38 (1977) 10–14. doi:10.1103/PhysRevLett.38.10.
[7] D. J. Rowe, Microscopic theory of the nuclear collective model, Rep. Prog. Phys. 48 (1985) 1419–1480. doi:10.1088/0034-4885/48/10/003.
[8] D. J. Rowe, Nuclear shape coexistence from the perspective of an algebraic many-nucleon version of the Bohr-Mottelson unified model, Phys. Rev. C 101 (2020) 054301. doi:10.1103/PhysRevC.101.054301.
[9] F. Pan, S. Yuan, K. D. Launey, J. P. Draayer, A new procedure for constructing basis vectors of SU(3) ⊃ SO(3), Nucl. Phys. A 952 (2016) 70. doi:10.1016/j.nuclphysa.2016.04.024.
[10] D. J. Rowe, C. Bahri, Clebschgordan coefficients of SU(3) in SU(2) and SO(3) basis, J. Math. Phys. 41 (2004) 6544. doi:10.1063/1.1286768.
[11] A. C. N. Martins, M. W. Suffak, H. de Guise, SU(3) Clebsch-Gordan coefficients and some of their symmetries, J. Phys. A: Math. Theor. 53 (2019) 025201. doi:10.1088/1751-8121/ab4b70.
[12] Y. Akiyama, J. P. Draayer, Users guide to fortran programs for wigner and racah coefficients of su3, Comp. Phys. Comm. 5 (1973) 405–415. doi:10.1016/0010-4655(73)90077-5.
[13] C. Bahri, D. J. Rowe, J. P. Draayer, Programs for generating clebschgordan coefficients of SU(3) in SU(2) and SO(3) bases, Comp. Phys. Comm. 159 (2004) 121–143. doi:10.1016/j.cpc.2004.01.005.
[14] J. P. Draayer, Y. Akiyama, Wigner and racah coefficients of SU(3), J. Math. Phys. 14 (12) (1973) 1904–1912. doi:10.1063/1.1666267.
[15] T. Dytrych, K. Launey, J. Draayer, P. Maris, J. Vary, E. Saule, U. Catalyurek, M. Sosonkina, D. Langr, M. Caprio, Collective modes in light nuclei from first principles, Phys. Rev. Lett. 111 (25) (2013) 252501. doi:10.1103/PhysRevLett.111.252501.
[16] K. Launey, T. Dytrych, Draayer, Symmetry-guided large-scale shell-model theory, Prog. Part. Nucl. Phys. 89 (2016) 101. doi:10.1016/j.ppnp.2016.02.001.
[17] T. Dytrych, T. D. Launey, J. P. Draayer, D. J. Rowe, J. L. Wood, G. Rosensteel, C. Bahr, D. Langr, R. Baker, Physics of nuclei: Key role of an emergent symmetry, Phys. Rev. Lett. 124 (2020) 042501. doi:10.1103/PhysRevLett.124.042501.
[18] K. D. Launey, T. Dytrych, G. Sargsyan, R. B. Baker, J. P. Draayer, Emergent symplectic symmetry in atomic nuclei, Eur. Phys. J. Special Topics 229 (2020) 2429. doi:10.1140/ep jst/e2020-000178-3.
[19] R. B. Baker, K. D. Launey, S. Bacca, N. N. Dinur, T. Dytrych, Benchmark calculations of electromagnetic sum rules with a symmetry-adapted basis and hyperspherical harmonics, Phys. Rev. C 102 (2020) 014320. doi:10.1103/PhysRevC.102.014320.
[20] A. C. Dreyfuss, K. D. Launey, J. E. Escher, G. H. Sargsyan, R. B. Baker, T. Dytrych, J. P. Draayer, Clustering and α-capture reaction rate from ab initio symmetry-adapted descriptions of ^{20}ne, Phys. Rev. C 102 (2020) 044608. doi:10.1103/PhysRevC.102.044608.
[21] A. E. McCoy, M. A. Caprio, T. Dytrych, P. Fassano, Emergent Sp(3,R) dynamical symmetry in the nuclear many-body system from an *Ab Initio* description, Phys. Rev. Lett. 125 (2020) 102505. doi:10.1103/PhysRevLett.125.102505.
[22] H. T. Johansson, C. Forssén, Fast and accurate evaluation of wigner 3j, 6j, and 9j symbols using prime factorization and multiword integer arithmetic, SIAM J. Sci. Comput. 38 (1) (2016) A376–A384. doi:10.1137/15M1021908.
[23] D. J. Millener, A note on recoupling coefficients for SU(3), J. Math. Phys. 19 (7) (1978) 1513–1514. doi:10.1063/1.523858.
[24] A. C. Dreyfuss, K. D. Launey, T. Dytrych, J. P. Draayer, C. Bahri, Hoyle state and rotational features in Carbon-12 within a no-core shell model framework, Phys. Lett. B 727 (2013) 511.